ACTIVE SUBSPACE METHODS IN THEORY AND PRACTICE: APPLICATIONS TO KRIGING SURFACES

PAUL G. CONSTANTINE*, ERIC DOW†, AND QIQI WANG‡

Abstract. Many multivariate functions encountered in practical engineering models vary primarily along a few directions in the space of input parameters. When these directions correspond to coordinate directions, one may apply global sensitivity measures to determine the most influential parameters. However, these methods perform poorly when the directions of variability are not aligned with the natural coordinates of the input space. We present a method to first detect the directions of variability using evaluations of the gradient and subsequently exploit these directions to construct a response surface on a low dimensional subspace – i.e., the *active subspace* – of the inputs. We develop a rigorous theoretical framework with computable error bounds, and we link the theoretical quantities to the parameters of a kriging response surface on the active subspace. We apply the method to a nonlinear heat transfer model on a turbine blade with a 250-parameter heat flux model representing uncertain transition to turbulence, and we compare its performance to other common approaches for input dimension reduction.

Key words. active subspace methods, dimension reduction, high dimensional approximation, kriging, response surfaces

1. Introduction & motivation.

As computational models of physical systems become more complex, the need increases for uncertainty quantification (UQ) to enable defensible predictions. Monte Carlo methods are the workhorse of UQ, where model inputs are sampled according to a characterization of their uncertainty and the corresponding model outputs are treated as a data set for statistical analysis. However, the slow convergence of Monte Carlo methods coupled with the high computational cost of the models has led many to employ response surfaces trained on a few carefully selected runs in place of the full model. This strategy has had great success in forward and inverse uncertainty propagation problems [24, 28, 10] as well as optimization [21]. However, most response surfaces suffer from the curse of dimensionality, where the cost of constructing an accurate surface increases exponentially as the dimension (i.e., the number of input parameters) increases.

To make construction tractable, one may first perform sensitivity analysis [35] to determine which variables have the most influence on the model predictions. With a ranking of the inputs, one may construct response surfaces that focus on the most influential variables, e.g., through a suitably anisotropic design; the same concept applies to mesh refinement strategies for solving PDEs. Methods for sensitivity analysis are typically classified as local perturbation or global methods. Local methods perturb one input at a time around a nominal value and measure the effects on the outputs. These methods are relatively inexpensive but lack robustness in the presence of noisy data. Local methods are fraught with other difficulties, e.g., how much should one perturb the input? And how can one be sure that the effects measured at the nominal condition are similar elsewhere in the parameter space? Global methods address these issues by providing integrated measures of the output's variability over the full range of parameters; consequently, they are computationally more expensive. Methods based on variance decompositions [30, 35] require approximating

*Stanford University, Stanford, California 94305 (paul.constantine@stanford.edu).

†Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139 (qiqi@mit.edu).

‡Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139 (qiqi@mit.edu).

high-dimensional integrals in order to rank the inputs.

Both classes of methods rank the coordinates of the inputs. However, some models may vary primarily along directions of the input space that are not aligned with the coordinate system. The analogy of inputs as knobs is useful here; turning two knobs simultaneously in some relative proportion may have a greater effect than turning only the most influential knob. Consider the function $\exp(ax_1 + bx_2)$ shown in Figure 5.1 with particular values of a and b. Changing (x_1, x_2) proportional to (a, b) will have the greatest effect, while changing the variables proportional to $(b, -a)$ will have no effect on the function. In effect, the function has one direction of variability and an orthogonal direction of flatness. It would be useful to know such directions for a given model with many inputs, where we could construct a response surface only along the directions of variability, i.e., on a subspace of model inputs.

We propose a method based on gradient evaluations for detecting and exploiting the directions of variability of a given function to construct an approximation on a low-dimensional subspace of the function's inputs. Given continued interest in gradient computations based on adjoint methods [5, 19] and algorithmic differentiation [16], it is not unreasonable to assume that one has access to the gradient of the function. We detect the directions by evaluating the function's gradient at a set of input points and determining a rotation of the input space that separates the directions of relative variability from directions of relative flatness. We exploit these directions by first projecting the input space to the low-dimensional subspace that captures the function's variability and then approximating the function on the subspace. Following Russi's 2010 Ph.D. thesis [33], we call this low-dimensional subspace the *active subspace*.

Subspace approximations are commonly used in optimization, where local quadratic models of a function are decomposed to reveal search directions. They are also found in many areas of model reduction [2] and optimal control [38], where a high-dimensional state space vector is approximated by a linear combination of relatively few basis vectors. Common to both of these fields are methods for matrix factorizations and eigenvalue computations [34], which are replete with subspace oriented approaches. The use of subspace methods for approximating high-dimensional functions arising in science and engineering models appears rare by comparison. Recent work by Lieberman et al [25] describes a method for finding a subspace in a high-dimensional input space via a greedy optimization procedure. Russi [33] proposes a method for discovering the *active subspace* of a function using evaluations of the gradient and constructing a quadratic response surface on the subspace; his methodology is similar to ours. Recently Fornasier et al [13] analyzed subspace approximation algorithms that do not need gradient evaluations but make strong assumptions on the function they are approximating; they take advantage of results from compressed sensing. Our previous work has applied the active subspace method to design optimization [7, 12], inverse analysis [10], and spatial sensitivity [11].

The contribution of this paper is two-fold. First we provide a rigorous theoretical foundation for gradient-based dimension reduction and subspace approximation. We construct and factorize a covariance-like matrix of the gradient to determine the directions of variability. We then construct a low-dimensional best approximation of the function via conditional expectation. We derive an error bound for the approximation in terms of the eigenvalues of the gradient covariance matrix along with a corollary that justifies our computational procedure. Second, we provide a bridge between the theoretical analysis and computational practice by (i) relating the derived error bounds to SVD-based approaches for discovering the active subspace and (ii)

heuristically linking the theoretical quantities to the parameters of a kriging response surface constructed on the active subspace. We apply this procedure to a nonlinear heat transfer model on a turbine blade, where a 250 parameter model for heat flux on the blade surface represents an uncertain transition zone from laminar to turbulent flow. We compare the subspace dimension reduction to other common strategies for sensitivity-based dimension reduction.

2. Theoretical framework. In this section we construct the theoretical framework for approximating a multivariate function on the active subspace including error bounds. We perform the analysis using techniques from probability theory, but we emphasize that there is nothing inherently stochastic about the functions or the approximations. We will use a subscript to denote conditional expectation. For example, if $f = f(\mathbf{y}, \mathbf{z})$, then we use $\mathbb{E}_{\mathbf{y}}[f]$ to denote the conditional expectation of f given $\mathbf{y}$. We will also use the subscript for the gradient operator to denote partial derivatives with respect to particular variables.

2.1. Directions of variability and the active subspace. Consider the m-dimensional function

$$f \;=\; f(\mathbf{x}), \quad \mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^m. \tag{2.1}$$

Let the space $\mathcal{X}$ be equipped with a probability density function $\rho = \rho(\mathbf{x})$. We assume that f is continuous, differentiable, and square integrable with respect to ρ. Denote the gradient of f by the column vector

$$\nabla_{\mathbf{x}} f(\mathbf{x}) \;=\; \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_m} \end{bmatrix}. \tag{2.2}$$

Define the $m \times m$ matrix $\mathbf{C}$ by

$$\mathbf{C} \;=\; \mathbb{E}\left[(\nabla_{\mathbf{x}} f)\,(\nabla_{\mathbf{x}} f)^T\right], \tag{2.3}$$

where we assume that f is such that $\mathbf{C}$ exists. $\mathbf{C}$ can be interpreted as the uncentered covariance of the gradient vector. Note that $\mathbf{C}$ is symmetric and positive semidefinite, so it admits a real eigenvalue decomposition

$$\mathbf{C} \;=\; \mathbf{W}\Lambda\mathbf{W}^T, \quad \Lambda \;=\; \operatorname{diag}(\lambda_1, \ldots, \lambda_m), \quad \lambda_1 \geq \cdots \geq \lambda_m \geq 0. \tag{2.4}$$

The following lemma quantifies the relationship between the gradient of f and the eigendecomposition of $\mathbf{C}$.

LEMMA 2.1. *The mean squared gradient of f along the direction given by the eigenvector $\mathbf{w}_i$ is equal to the corresponding eigenvalue,*

$$\mathbb{E}\left[((\nabla_{\mathbf{x}} f)^T \mathbf{w}_i)^2\right] \;=\; \lambda_i \tag{2.5}$$

Proof. By the definition of $\mathbf{C}$,

$$\lambda_i \;=\; \mathbf{w}_i^T \mathbf{C} \mathbf{w}_i \;=\; \mathbf{w}_i^T \left(\mathbb{E}\left[(\nabla_{\mathbf{x}} f)\,(\nabla_{\mathbf{x}} f)^T\right]\right) \mathbf{w}_i \;=\; \mathbb{E}\left[((\nabla_{\mathbf{x}} f)^T \mathbf{w}_i)^2\right], \tag{2.6}$$

as required. ∎

Lemma 2.1 implies, for example, that if λ_i is zero, then f is flat along the direction $\mathbf{w}_i$ in $\mathbb{R}^m$. The essence of the low dimensional approximation is to detect and remove directions along which f varies relatively little.

The eigenvectors $\mathbf{W}$ define a rotation of $\mathbb{R}^m$ and consequently the domain of f. With eigenvalues in decreasing order, we can separate components of the rotated coordinate system into a space which contains the majority of f's variability – i.e., the *active subspace* – and its orthogonal complement along which f is on average relatively flat. The eigenvalues and eigenvectors are partitioned as

$$\Lambda = \begin{bmatrix} \Lambda_1 & \\ & \Lambda_2 \end{bmatrix}, \qquad \mathbf{W} = \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 \end{bmatrix} \tag{2.7}$$

where $\Lambda_1 = \text{diag}\,(\lambda_1, \ldots, \lambda_n)$ with $n < m$, and $\mathbf{W}_1$ is $m \times n$. Define the rotated coordinates $[\mathbf{y}, \mathbf{z}]^T$ by

$$\mathbf{y} = \mathbf{W}_1^T \mathbf{x}, \quad \mathbf{z} = \mathbf{W}_2^T \mathbf{x}. \tag{2.8}$$

Then we have the following lemma.

LEMMA 2.2. *The mean squared gradients of f with respect to the coordinates $[\mathbf{y}, \mathbf{z}]^T$ satisfy*

$$\begin{aligned} \mathbb{E}\left[(\nabla_{\mathbf{y}} f)^T (\nabla_{\mathbf{y}} f)\right] &= \lambda_1 + \cdots + \lambda_n \\ \mathbb{E}\left[(\nabla_{\mathbf{z}} f)^T (\nabla_{\mathbf{z}} f)\right] &= \lambda_{n+1} + \cdots + \lambda_m \end{aligned} \tag{2.9}$$

Proof. First note that we can write

$$\begin{aligned} f(\mathbf{x}) &= f(\mathbf{W}\mathbf{W}^T\mathbf{x}) \\ &= f(\mathbf{W}_1\mathbf{W}_1^T\mathbf{x} + \mathbf{W}_2\mathbf{W}_2^T\mathbf{x}) \\ &= f(\mathbf{W}_1\mathbf{y} + \mathbf{W}_2\mathbf{z}) \end{aligned} \tag{2.10}$$

By the chain rule, the gradient of f with respect to $\mathbf{y}$ can be written

$$\nabla_{\mathbf{y}} f(\mathbf{x}) = \nabla_{\mathbf{y}} f(\mathbf{W}_1\mathbf{y} + \mathbf{W}_2\mathbf{z}) = \mathbf{W}_1^T \nabla_{\mathbf{x}} f(\mathbf{W}_1\mathbf{y} + \mathbf{W}_2\mathbf{z}) = \mathbf{W}_1^T \nabla_{\mathbf{x}} f(\mathbf{x}). \tag{2.11}$$

Then

$$\begin{aligned} \mathbb{E}\left[(\nabla_{\mathbf{y}} f)^T (\nabla_{\mathbf{y}} f)\right] &= \mathbb{E}\left[\text{trace}\left((\nabla_{\mathbf{y}} f)(\nabla_{\mathbf{y}} f)^T\right)\right] \\ &= \text{trace}\left(\mathbb{E}\left[(\nabla_{\mathbf{y}} f)(\nabla_{\mathbf{y}} f)^T\right]\right) \\ &= \text{trace}\left(\mathbf{W}_1^T \mathbb{E}\left[(\nabla_{\mathbf{x}} f)(\nabla_{\mathbf{x}} f)^T\right] \mathbf{W}_1\right) \\ &= \text{trace}\left(\mathbf{W}_1^T \mathbf{C} \mathbf{W}_1\right) \\ &= \text{trace}\left(\Lambda_1\right) \\ &= \lambda_1 + \cdots + \lambda_n, \end{aligned} \tag{2.12}$$

as required. The derivation for the $\mathbf{z}$ components is similar. ∎

Lemma 2.2 motivates the use of the label *active subspace*. In particular, f varies more on average in the subspace defined by the columns of $\mathbf{W}_1$ than in the subspace defined by $\mathbf{W}_2$, as quantified by the eigenvalues of $\mathbf{C}$.

2.1.1. Two important special cases. There are two cases that show up in practice where the rank of $\mathbf{C}$ may be determined a priori. The first is a *ridge function* [8], which has the form

$$f(\mathbf{x}) \;=\; h(\mathbf{a}^T\mathbf{x}), \tag{2.13}$$

where h is a univariate function, and $\mathbf{a}$ is a constant m-vector. In this case, $\mathbf{C}$ is rank one, and the eigenvector defining the active subspace is $\mathbf{a}/\|\mathbf{a}\|$, which can be discovered by a single evaluation of the gradient anywhere in $\mathcal{X}$. The function shown in Figure 5.1 is an example of a ridge function.

The second special case is a function of the form

$$f(\mathbf{x}) \;=\; h(\mathbf{x}^T\mathbf{A}\mathbf{x}), \tag{2.14}$$

where h is a univariate function and $\mathbf{A}$ is a symmetric $m \times m$ matrix. In this case

$$\mathbf{C} \;=\; \mathbf{A}\,\mathbb{E}\left[(h'(\mathbf{x}^T\mathbf{A}\mathbf{x}))^2\,\mathbf{x}\mathbf{x}^T\right]\,\mathbf{A}^T, \tag{2.15}$$

where h' is the derivative of h. This implies that the null space of $\mathbf{C}$ is the null space of $\mathbf{A}$ provided that h' is non-degenerate.

2.2. Approximation on the active subspace. We can now construct a function of n variables that approximates f. Before formally defining such a function, we offer some intuition based on the decomposition in (2.10). Since perturbations in $\mathbf{z}$ change f relatively little on average, we would like to have a function of only $\mathbf{y}$ that approximates f. For a fixed $\mathbf{y}$, the best guess one could make for f is the mean of f over all values of $\mathbf{z}$. This is precisely the conditional expectation of f given $\mathbf{y}$. Define the n-variate function $f_n(\mathbf{y})$ by

$$f_n \;=\; f_n(\mathbf{y}) \;=\; \mathbb{E}_{\mathbf{y}}\left[f\right]. \tag{2.16}$$

Since f_n is a conditional expectation, it is the best approximation of f given $\mathbf{y}$ [37, Chapter 9]. The domain of f_n is

$$\mathcal{Y} \;=\; \left\{\mathbf{y} \;:\; \mathbf{y} = \mathbf{W}_1^T\mathbf{x},\; \mathbf{x} \in \mathcal{X}\right\}, \tag{2.17}$$

The next theorem bounds the error of f_n in terms of the eigenvalues of $\mathbf{C}$ from (2.3).

THEOREM 1. *The mean squared error of f_n defined in* (2.16) *is bounded by*

$$\mathbb{E}\left[(f - f_n)^2\right] \;\leq\; C_P\left(\lambda_{n+1} + \cdots + \lambda_m\right) \tag{2.18}$$

where C_P is a constant that depends only on the domain $\mathcal{X}$ and the weight function ρ.

Proof. Note that $\mathbb{E}_{\mathbf{y}}\left[f - f_n\right] = 0$ by the definition (2.16). Thus,

$$\begin{aligned}
\mathbb{E}\left[(f - f_n)^2\right] &= \mathbb{E}\left[\mathbb{E}_{\mathbf{y}}\left[(f - f_n)^2\right]\right] && (2.19)\\
&\leq C_P\,\mathbb{E}\left[\mathbb{E}_{\mathbf{y}}\left[(\nabla_{\mathbf{z}} f)^T(\nabla_{\mathbf{z}} f)\right]\right] && (2.20)\\
&= C_P\,\mathbb{E}\left[(\nabla_{\mathbf{z}} f)^T(\nabla_{\mathbf{z}} f)\right] && (2.21)\\
&= C_P\left(\lambda_{n+1} + \cdots + \lambda_m\right). && (2.22)
\end{aligned}$$

Lines (2.19) and (2.21) are due to the tower property of conditional expectations. Line (2.20) is a Poincare inequality, where the constant C_P depends only on $\mathcal{X}$ and the density function ρ. Line (2.22) follows from Lemma 2.2. ∎

Theorem 1 provides an error bound for the approximation f_n. However, f_n may be very difficult to compute in practice since every evaluation given $\mathbf{y}$ requires a high dimensional integral over the coordinates $\mathbf{z}$. We would like to use a single evaluation of $f(\mathbf{x})$ to approximate $f_n(\mathbf{y})$ with $\mathbf{y} = \mathbf{W}_1^T\mathbf{x}$. In the following corollary, we show that such a strategy is well motivated if the variation of f along the coordinates $\mathbf{z}$ is sufficiently small.

COROLLARY 2.3. *Given $\varepsilon > 0$, the probability that f deviates from f_n by ε is bounded by*

$$\mathbb{P}\left[|f - f_n| \geq \varepsilon\right] \;\leq\; \frac{C_P}{\varepsilon^2}\left(\lambda_{n+1} + \cdots + \lambda_m\right) \tag{2.23}$$

where C_P is the Poincare constant from Theorem 1.

Proof. This follows directly from the Chebyshev inequality and Theorem 1. ∎

3. Computational framework. In this section we present a computational framework for constructing an approximation to $f_n(\mathbf{y}) = \mathbb{E}_{\mathbf{y}}\left[f\right]$; section 3.3 contains a step-by-step computational algorithm, and eager readers may begin there. We assume that it is possible to evaluate the gradient $\nabla_{\mathbf{x}} f$ at a given point $\mathbf{x} \in \mathcal{X}$. This is not an unreasonable assumption given the increased interest in adjoint methods for computing sensitivities [19, 5] and algorithmic differentiation [16]. If derivatives are not available, then one can in principle use finite differences to evaluate the gradient. However, this approach requires $m + 1$ evaluations of f for every evaluation of the gradient, which may be infeasible in many applications. Also, finite difference methods fail when applied to noisy function evaluations. An intriguing alternative can be found in [13], where derivatives are not required but the function must have special structure.

3.1. Discovering the active subspace. We must first compute the eigenvectors $\mathbf{W}$ and eigenvalues Λ of the matrix $\mathbf{C}$ from (2.4). We immediately encounter the obstacle of computing the elements of $\mathbf{C}$, which are integrals over the high dimenisonal space $\mathcal{X}$. Thus, tensor product numerical quadrature rules are impractical. We opt for Monte Carlo integration, which will yield its own appealing interpretation. In particular, let

$$\nabla_{\mathbf{x}} f_j \;=\; \nabla_{\mathbf{x}} f(\mathbf{x}_j), \qquad \mathbf{x}_j \in \mathcal{X}, \quad j = 1, \ldots, M, \tag{3.1}$$

be independently computed samples of the gradient vector, where $\mathbf{x}_j$ is drawn from the density ρ on $\mathcal{X}$. Note that in practice computing the gradient at $\mathbf{x}_j$ typically involves first evaluating $f_j = f(\mathbf{x}_j)$; we will use these function evaluations when training and testing the response surface. With the samples of the gradient, we approximate

$$\mathbf{C} \;\approx\; \tilde{\mathbf{C}} \;=\; \frac{1}{M}\sum_{j=1}^{M} (\nabla_{\mathbf{x}} f_j)(\nabla_{\mathbf{x}} f_j)^T, \tag{3.2}$$

and compute the eigenvalue decomposition

$$\tilde{\mathbf{C}} \;=\; \tilde{\mathbf{W}}\tilde{\Lambda}\tilde{\mathbf{W}}^T. \tag{3.3}$$

The size of $\tilde{\mathbf{C}}$ is $m \times m$, where we expect m to be on the order of hundreds or thousands corresponding to the number of variables $\mathbf{x}$. Thus we anticipate no issues computing the complete eigendecomposition of $\tilde{\mathbf{C}}$ on a modern personal computer.

There is another interpretation of the sampling approach to approximate the eigenpairs of $\mathbf{C}$. We can write

$$\tilde{\mathbf{C}} \;=\; \mathbf{G}\mathbf{G}^T, \tag{3.4}$$

where the $m \times M$ matrix $\mathbf{G}$ is

$$\mathbf{G} \;=\; \frac{1}{\sqrt{M}} \left[\nabla_{\mathbf{x}} f_1 \quad \cdots \quad \nabla_{\mathbf{x}} f_M \right]. \tag{3.5}$$

If we compute the singular value decomposition (SVD) $\mathbf{G} = \mathbf{U}\Sigma\mathbf{V}^T$, where $\mathbf{U}$ has dimension $m \times m$ and $\mathbf{V}$ has dimension $M \times m$, then

$$\tilde{\mathbf{C}} \;=\; \mathbf{G}\mathbf{G}^T \;=\; \left(\mathbf{U}\Sigma\mathbf{V}^T\right)\left(\mathbf{V}\Sigma^T\mathbf{U}^T\right) \;=\; \mathbf{U}\Sigma^2\mathbf{U}^T. \tag{3.6}$$

Thus, $\mathbf{U} = \tilde{\mathbf{W}}$ up to a change in sign, and $\Sigma^2 = \tilde{\Lambda}$. This provides an alternative computational approach via the SVD. Again, we stress that the number of variables m and the number of gradient samples M are sufficiently small in many applications of interest so that the SVD can easily be computed on a modern personal computer. More importantly, the SVD shows that the rotation matrix $\tilde{\mathbf{W}}$ can be interpreted as the uncentered principal directions [20] from an ensemble of gradient evaluations.

In practice, we use $\tilde{\mathbf{W}}$ and $\tilde{\Lambda}$ in place of $\mathbf{W}$ and Λ. We will omit the distinction between the two in the remainder of the paper. The errors in this approximation may be significant, particularly if the number M of samples of the gradient is too small. However, a thorough analysis of these errors is beyond the scope of the present work. We anticipate that results from classical perturbation theory for eigenvalues and eigenvectors [29] will prove useful.

3.2. Building a response surface. In this section, we detail a procedure to construct a response surface on the n-dimensional active subspace defined by $\mathbf{W}_1$ from (2.7). At this point, we assume that we have the eigenvectors $\mathbf{W}$ and eigenvalues Λ from the Monte Carlo estimate of $\mathbf{C}$ described in the previous section.

We must first choose the dimension n of the subspace. Many reduction methods use the magnitude of the λ_i to define n, e.g., so that $\lambda_1 + \cdots + \lambda_n$ exceeds some proportion of $\lambda_1 + \cdots + \lambda_m$. This, however, is not our aim. We are bound instead by more practical considerations, such as choosing n small enough to construct a reasonable set of training sites (e.g., a mesh) for the response surface; the trailing eigenvalues $\lambda_{n+1}, \ldots, \lambda_m$ then inform a noise model for the response surface. A rapid decay in the λ_i implies that the low dimensional approximation is relatively more accurate (see Theorem 1), but it is not necessary to apply the reduction method.

On the reduced domain $\mathcal{Y}$ in (2.17), we seek a function $\tilde{f}_n \approx f_n$, where f_n is the conditional expecation defined in (2.16). Global or piecewise polynomial approximation offers one possible construction. In this work, we will use a kriging response surface [22] (also known as Gaussian process approximation [32] and closely related to radial basis approximation [36]). The primary reason for this choice is that the computed λ_i can be used to inform the parameters of the kriging surface – both a variance model for the training data and the correlation lengths along each of the reduced coordinates. We demonstrate this in Section 3.2.3.

Once the function $\tilde{f}_n$ is trained, the following procedure is used to approximate f.

1. Given $\mathbf{x} \in \mathcal{X}$, compute $\mathbf{y} = \mathbf{W}_1^T\mathbf{x}$.
2. Evaluate the response surface $\tilde{f}_n(\mathbf{y})$.

3. Set $f(\mathbf{x}) = \tilde{f}_n(\mathbf{y})$.

In the context of kriging, $\tilde{f}_n(\mathbf{y})$ is interpreted as the expectation of a Gaussian random variable, and the variance of the prediction is provided as well.

3.2.1. The reduced domain. The domain of $f_n(\mathbf{y})$ is $\mathcal{Y}$ from (2.17). If the domain $\mathcal{X}$ of $f(\mathbf{x})$ is unbounded and equal to $\mathbb{R}^m$, then the reduced domain $\mathcal{Y}$ is also unbounded and equal to $\mathbb{R}^n$. In such a case, constructing the reduced domain is straightforward; it is simply a rotation followed by a coordinate projection applied to $\mathcal{X}$. An important special case is when $\mathcal{X}$ is unbounded and the weight function $\rho(\mathbf{x})$ is a Gaussian probability density function. In this case, the projected density function is also Gaussian. We do not pursue the case of unbounded domains further. However, the techniques we use to construct the response surface on the subspace for bounded domains are similar for unbounded domains.

If $\mathcal{X}$ is compact and $\rho(\mathbf{x})$ has compact support, then one must take care when constructing the reduced domain $\mathcal{Y}$. We will specialize to the case where $\mathcal{X}$ is a hypercube $[-1,1]^m$, which we write as

$$\mathcal{X} \ = \ \{\mathbf{x} \ : \ -1 \leq \mathbf{x} \leq 1\}. \tag{3.7}$$

Mathematically, this restriction is not necessary. But it is much more convenient computationally, since finding a point in $\mathcal{X}$ that corresponds to a given point in $\mathcal{Y}$ becomes a standard linear program [27].

When $\mathcal{X}$ is a hypercube, the reduced domain $\mathcal{Y}$ can be found by first projecting each corner of $\mathcal{X}$ to $\mathbb{R}^n$ by $\mathbf{W}_1$ and taking the convex hull. We write this as

$$\mathcal{Y} \ = \ \text{conv}\left(\left\{\mathbf{y} \ : \ \mathbf{y} = \mathbf{W}_1^T\mathbf{x}, \ \mathbf{x} \in \{-1,1\}^m\right\}\right), \tag{3.8}$$

where $\{-1,1\}^m$ is the space of m-vectors with entries equal to -1 or 1. Figure 5.2a shows the two dimensional projection of a three dimensional cube (blue) and the convex hull of its corners (green). However, this approach requires finding the convex hull of 2^m points in $\mathbb{R}^n$, which may be intractable if m is too large.

Instead, we propose to create a bounding box for $\mathcal{Y}$ as follows. For each column $\mathbf{w}_i$ from $\mathbf{W}_1$, compute upper and lower bounds for the corresponding reduced variable y_i with a bound constrained linear program

$$y_i^l \ = \ \underset{-1\leq\mathbf{x}\leq 1}{\text{minimum}}(\mathbf{w}_i^T\mathbf{x}), \qquad y_i^u \ = \ -y_i^l. \tag{3.9}$$

In fact, the linear program has a closed form solution $y_i^l = -\mathbf{w}_i^T \,\text{sign}(\mathbf{w}_i)$, where $\text{sign}(\mathbf{w}_i)$ returns an m-vector with the sign of each component of $\mathbf{w}_i$. However, we find the linear program formulation more intuitive.

The bounding box is then given by

$$\bar{\mathcal{Y}} \ = \ \{\mathbf{y} \ : \ \mathbf{y} \in \mathbb{R}^n, \ \ \mathbf{y}_l \leq \mathbf{y} \leq \mathbf{y}_u\}, \tag{3.10}$$

where $\mathbf{y}_l = [y_1^l, \ldots, y_n^l]^T$ and $\mathbf{y}_u = [y_1^u, \ldots, y_n^u]^T$. By construction $\mathcal{Y} \subseteq \bar{\mathcal{Y}}$. Figure 5.2a shows the two dimensional bounding box (red) for the projected three dimensional cube (blue).

3.2.2. Training data on the reduced domain. To create training data for the kriging surface on the active subspace, we must evaluate the function $\tilde{f}_n$ at a set of points in the bounding box domain $\bar{\mathcal{Y}}$. We already have M evaluations f_j that exist at the points $\mathbf{y}_j = \mathbf{W}_1^T\mathbf{x}_j$, which were computed during the sampling of the

gradient (Section 3.1). It is natural to want to use the f_j when training the kriging surface on the subspace to avoid unnecessary evaluations of f. However, there is no guarantee that the points $\mathbf{y}_j$ comprise a good design on the subspace; they will likely cluster together, which causes difficulties for the conditioning of the kriging surface. Therefore, we set the existing function evaluations aside during the training and use them instead to test the quality of the response surface.

To ensure a good design on the reduced domain, we choose it independently of the $\mathbf{x}_j$ used to sample the gradient of f. Let $\mathbf{y}_k$ with $k = 1, \ldots, N$ be an initial set of design points in the bounding box domain $\bar{\mathcal{Y}}$ chosen to enable a well-conditioned training step. There are many options for choosing the points of an experimental design, many of which are detailed in [22]. In Figure 5.2a, we show the points of a tensor grid of uniformly spaced points in red as an example.

For each $\mathbf{y}_k$, we want to find a point $\mathbf{x}^*$ in the full m-dimensional domain $\mathcal{X}$ and set the training data equal to $f(\mathbf{x}^*)$. However, not every design site on the bounding box $\bar{\mathcal{Y}}$ will correspond to a point in $\mathcal{X}$. We must therefore remove sites from the initial design on $\bar{\mathcal{Y}}$ that do not correspond to points in $\mathcal{X}$. For the hypercube case, this involves solving Phase 1 of a linear program [27] for each point in the design. In particular, for $\mathbf{y}_k \in \bar{\mathcal{Y}}$, we seek $\mathbf{x}^*$ that satisfies

$$\mathbf{y}_k \;=\; \mathbf{W}_1^T \mathbf{x}^*, \qquad -1 \leq \mathbf{x}^* \leq 1. \tag{3.11}$$

If no such $\mathbf{x}^*$ exists, then the linear program solver will indicate that the constraint set is infeasible. In this case we remove $\mathbf{y}_k$ from the initial design and decrement N by one. We are left with a subset of the initial design sites where we are guaranteed that each remaining site corresponds to at least one point in $\mathcal{X}$. The initial design on $\bar{\mathcal{Y}}$ is shown in Figure 5.2b; the remaining points after pruning are shown in Figure 5.2c with the true reduced domain in green for reference.

For each remaining $\mathbf{y}_k$, we find $\mathbf{x}^*$ that satisfies (3.11) and compute the training data

$$\tilde{f}_{n,k} \;=\; \tilde{f}_n(\mathbf{y}_k) \;=\; f(\mathbf{x}^*). \tag{3.12}$$

There may be an infinite number of such $\mathbf{x}^*$, each corresponding to a different value of the coordinates $\mathbf{z}$. By Corollary 2.3, we are assured that the value $f(\mathbf{x}^*)$ is unlikely to deviate too far from its conditional mean $f_n(\mathbf{y}_k)$. Motivated by the error bounds in Theorem 1, we will build a noise model that represents the deviation of $f(\mathbf{x}^*)$ from $f_n(\mathbf{y}_k)$. And we will use this noise model when constructing a kriging surface.

3.2.3. Training the kriging response surface. At this point we have values $\tilde{f}_{n,k}$ for each design site $\mathbf{y}_k \in \bar{\mathcal{Y}}$ that we will use to train the kriging surface. We assume that the function we are trying to approximate is smooth with respect to the coordinates $\mathbf{y}$. However, since the training data $f_{n,k}$ are not exactly equal to the conditional expectation $f_n(\mathbf{y}_k)$, we do not want to force the kriging surface to exactly interpolate the training data. Instead, we want to build a model for the noise in the training data that is motivated by Theorem 1 and incorporate it into the kriging surface. We therefore choose the correlation matrix of the training data to be a product-type squared exponential kernel with an additional diagonal term to represent the noise,

$$\mathrm{Cov}\left[\tilde{f}_n(\mathbf{y}_{k_1}),\, \tilde{f}_n(\mathbf{y}_{k_2})\right] \;=\; K\left(\mathbf{y}_{k_1}, \mathbf{y}_{k_2}\right) + \eta^2\, \delta(k_1, k_2) \tag{3.13}$$

where $\delta(k_1, k_2)$ is 1 if $k_1 = k_2$ and zero otherwise, and

$$K\left(\mathbf{y}_{k_1}, \mathbf{y}_{k_2}\right) \;=\; \exp\left(-\sum_{i=1}^{n} \frac{\left(y_{k_1,i} - y_{k_2,i}\right)^2}{2\ell_i^2}\right). \tag{3.14}$$

Along with the correlation function, we choose a quadratic mean term, which will add $\binom{n+2}{n}$ polynomial basis functions to the kriging approximation.

We are left to determine the parameters of the kriging surface, including the correlation lengths ℓ_i from (3.14) and the parameter η^2 of the noise model from (3.13). Given values for these parameters, the coefficients of both the polynomial bases and the linear combination of the training data are computed in the standard way [22, 32]. We could also use a standard maximum likelihood method to compute ℓ_i and η^2. However, we can inform these parameters using the quantities from Lemma 2.1 and Theorem 1.

Toward this end, we approximate the directional derivative of f along $\mathbf{w}_i$ with a finite difference,

$$\nabla_{\mathbf{x}} f(\mathbf{x})^T \mathbf{w}_i \;\approx\; \frac{1}{\delta}\left(f(\mathbf{x} + \delta\mathbf{w}_i) - f(\mathbf{x})\right), \tag{3.15}$$

which is valid for small δ. Decompose $f(\mathbf{x}) = f_0 + f'(\mathbf{x})$, where $f_0 = \mathbb{E}\left[f\right]$, so that f' has zero-mean. By Lemma 2.1,

$$\begin{aligned}
\lambda_i &= \mathbb{E}\left[\left((\nabla_{\mathbf{x}} f)^T \mathbf{w}_i\right)^2\right] \\
&\approx \frac{1}{\delta^2}\mathbb{E}\left[\left(f(\mathbf{x} + \delta\mathbf{w}_i) - f(\mathbf{x})\right)^2\right] \\
&= \frac{1}{\delta^2}\mathbb{E}\left[\left(f'(\mathbf{x} + \delta\mathbf{w}_i) - f'(\mathbf{x})\right)^2\right] \\
&= \frac{2}{\delta^2}\left(\sigma^2 - \mathrm{Cov}\left[f'(\mathbf{x} + \delta\mathbf{w}_i),\, f'(\mathbf{x})\right]\right),
\end{aligned} \tag{3.16}$$

where $\sigma^2 = \mathrm{Var}\left[f\right]$. Rearranged, we have

$$\mathrm{Corr}\left[f'(\mathbf{x} + \delta\mathbf{w}_i),\, f'(\mathbf{x})\right] \;=\; \frac{1}{\sigma^2}\mathrm{Cov}\left[f'(\mathbf{x} + \delta\mathbf{w}_i),\, f'(\mathbf{x})\right] \;=\; 1 - \frac{\lambda_i \delta^2}{2\sigma^2}. \tag{3.17}$$

This implies that the correlation function along $\mathbf{w}_i$ is locally quadratic near the origin with coefficient $\lambda_i / 2\sigma^2$. A univariate Gaussian correlation function with correlation length parameter ℓ has a Taylor series about the origin

$$\exp\left(-\frac{\delta^2}{2\ell^2}\right) \;=\; 1 - \frac{\delta^2}{2\ell^2} + \cdots \tag{3.18}$$

Comparing these terms to the locally quadratic approximation of the correlation function, we can use a univariate Gaussian with correlation length

$$\ell_i^2 \;=\; \frac{\sigma^2}{\lambda_i} \tag{3.19}$$

for the reduced coordinate y_i. Thus we need to approximate the variance σ^2.

Applying Theorem 1 with $n = 0$, we have

$$\sigma^2 \;=\; \mathrm{Var}\left[f\right] \;\leq\; C_P\left(\lambda_1 + \cdots + \lambda_m\right). \tag{3.20}$$

Unfortunately, the Poincare inequality is a notoriously loose bound, so we are reluctant to simply plug in an estimate of C_P to approximate σ^2. Instead, we posit that for some constant α,

$$\sigma^2 \;=\; \alpha\,(\lambda_1 + \cdots + \lambda_m), \tag{3.21}$$

where $\alpha \leq C_P$, and we can employ an estimate of C_P for the hypercube from [4],

$$C_P \;=\; \frac{2\sqrt{m}}{\pi}. \tag{3.22}$$

To get a rough lower bound on α, we use the biased estimator of σ^2 from the samples f_j computed in Section 3.1,

$$\hat{\sigma}^2 \;=\; \frac{1}{M}\sum_{j=1}^{M}(f_j - \hat{f}_0)^2, \tag{3.23}$$

where $\hat{f}_0$ is the empirical mean of the f_j. Since M will generally be small due to limited function evaluations, we expect the bias to be significant enough to justify the bound

$$\hat{\sigma}^2 \;\leq\; \alpha\,(\lambda_1 + \cdots + \lambda_m). \tag{3.24}$$

Combining (3.24) and (3.22), we have

$$\frac{\hat{\sigma}^2}{\lambda_1 + \cdots + \lambda_m} \;\leq\; \alpha \;\leq\; \frac{2\sqrt{m}}{\pi}. \tag{3.25}$$

From here, we treat α as a hyperparameter for the correlation kernel (3.14), and we can use a maximum likelihood approach to set it. Note that the number of variables in the maximum likelihood is one, in contrast to $n+1$ variables for the standard approach. Given a value for alpha, we set σ^2 by (3.21) and

$$\eta^2 \;=\; \alpha\,(\lambda_{n+1} + \cdots + \lambda_m) \tag{3.26}$$

in (3.13). We now have all necessary quantities to build the kriging surface on the active subspace.

3.3. A step-by-step algorithm. We summarize this section with an algorithm incorporating the previously described computational procedures given a function $f = f(\mathbf{x})$ and its gradient $\nabla_{\mathbf{x}} f = \nabla_{\mathbf{x}} f(\mathbf{x})$ defined on the hypercube $[-1,1]^m$.

1. **Initial sampling:** Choose a set of M points $\mathbf{x}_j \in [-1,1]^m$ according to the measure $\rho(\mathbf{x})$. For each $\mathbf{x}_j$, compute $f_j = f(\mathbf{x}_j)$ and $\nabla_{\mathbf{x}} f_j = \nabla_{\mathbf{x}} f(\mathbf{x}_j)$. Compute the sample variance $\hat{\sigma}^2$.
2. **Gradient analysis:** Compute the SVD of the matrix

$$\mathbf{G} \;=\; \frac{1}{\sqrt{M}}\begin{bmatrix}\nabla_{\mathbf{x}} f_1 & \cdots & \nabla_{\mathbf{x}} f_M\end{bmatrix} \;=\; \mathbf{W}\Sigma\mathbf{V}^T, \tag{3.27}$$

and set $\Lambda = \Sigma^2$. Choose a reduced dimension $n < m$ according to practical considerations and the decay of λ_i. Partition $\mathbf{W} = \begin{bmatrix}\mathbf{W}_1 & \mathbf{W}_2\end{bmatrix}$.

3. **Reduced domain:** Define the reduced coordinates $y_i = \mathbf{w}_i^T\mathbf{x}$ for $i = 1,\ldots,n$. Construct a bounding box for the reduced coordinates

$$y_i^l = \underset{-1\leq\mathbf{x}\leq 1}{\text{minimum}}(\mathbf{w}_i^T\mathbf{x}), \qquad y_i^u = -y_i^l, \tag{3.28}$$

with $\mathbf{y}_l = [y_1^l,\ldots,y_n^l]^T$ and $\mathbf{y}_u = [y_1^u,\ldots,y_n^u]^T$. Define the reduced domain

$$\bar{\mathcal{Y}} = \{\mathbf{y} : \mathbf{y}_l \leq \mathbf{y} \leq \mathbf{y}_u\}. \tag{3.29}$$

For points $\mathbf{x}_j$ in the initial design, define $\mathbf{y}_j = \mathbf{W}_1^T\mathbf{x}_j$. See Figures 5.2a-5.2c.
4. **Design on reduced domain:** Choose a set of N points $\mathbf{y}_k \in \bar{\mathcal{Y}}$. For each $\mathbf{y}_k$, use a linear program solver to find an $\mathbf{x}^* \in [-1,1]^m$ that satisfies $\mathbf{y}_k = \mathbf{W}_1^T\mathbf{x}^*$. If no such $\mathbf{x}^*$ exists, then remove $\mathbf{y}_k$ from the design.
5. **Compute training data:** For each remaining $\mathbf{y}_k$, set

$$\tilde{f}_{n,k} = \tilde{f}_n(\mathbf{y}_k) = f(\mathbf{x}^*). \tag{3.30}$$

6. **Train the response surface:** Use a maximum likelihood method to find the hyperparameter α with bounds

$$\frac{\hat{\sigma}^2}{\lambda_1+\cdots+\lambda_m} \leq \alpha \leq \frac{2\sqrt{m}}{\pi}. \tag{3.31}$$

Set the noise model parameter

$$\eta^2 = \alpha\left(\lambda_{n+1}+\cdots+\lambda_m\right) \tag{3.32}$$

and correlation lengths

$$\ell_i^2 = \frac{\alpha}{\lambda_i}\left(\lambda_1+\cdots+\lambda_m\right) \tag{3.33}$$

for the product squared exponential correlation kernel on $\bar{\mathcal{Y}}$. Apply standard kriging with the training data $\tilde{f}_{n,k}$.
7. **Evaluate the response surface:** For a point $\mathbf{x} \in [-1,1]^m$, compute $\mathbf{y} = \mathbf{W}_1^T\mathbf{x}$, and evaluate the kriging surface $\tilde{f}_n(\mathbf{y})$ and its prediction variance. Set $f(\mathbf{x}) = \tilde{f}_n(\mathbf{y})$.

We conclude this section with summarizing remarks. First, the case of an infinite domain is similar with fewer manipulations for the reduced domain, since there is no need for a bounding box. However, it is less clear how to construct a good design on an unbounded domain without artificial boundaries. Second, the methods proposed for the gradient analysis can be improved substantially with better Monte Carlo methods. Sequential sampling techniques combined with a measure of the stability of the computed subspaces could be a powerful approach for reducing the number of gradient evaluations. Alternatively, randomized algorithms for low rank approximation offer promise for reducing the number of gradient samples [6, 17].

Third, as written, the choice of n requires some interaction from the user. We advocate such interaction since we have found that one can uncover insights into the function f by examining the λ_i and the elements of $\mathbf{w}_i$. Fourth, as written, there is substantial freedom in choosing both the design sites on the reduced domain and the response surface. This was intentional. For our purposes, it suffices to use the gradient analysis to construct an approximation on the active subspace, but the

details of the approximation will require many more practical considerations than we can address here. We have chosen kriging primarily because of (i) the natural fit of the computed λ_i to the correlation length parameters and training data noise model, and (ii) its flexibility with scattered design sites. However, many other options for approximation are possible including global polynomials, regression splines, or finite element approximations; Russi advocates a global quadratic polynomial [33] on the subspace.

Fifth, with the gradient available for f, one could use (2.11) to obtain gradients with respect to the reduced coordinates. This could then be used to improve the response surface on the active subspace [22]. As also mentioned in [22], since we know a great deal about our correlation function, we could create designs that satisfy optimality criteria such as maximum entropy. However, this is not straightforward because the boundaries of the true reduced domain (not the bounding box) are difficult to set as constraints for an optimization procedure. And optimal designs on the bounding box are unlikely to be optimal on the true reduced domain.

Finally, we note that the function evaluations f_j could be better used to construct the response surface on the subspace. We have intentionally avoided proposing any strategies for such use and prefer instead to use them as a testing set for the response surface as detailed in the next section.

4. Numerical examples. In this section, we apply the dimension reduction method to a physically motivated model of heat transfer on a three-dimensional turbine blade with interior cooling holes and a parameterized model for the heat flux on the surface. The problem is motivated by [31], where the authors use a Reynolds averaged Navier-Stokes (RANS) model of high speed flow over a turbine blade to study transition to turbulence of the flow field and its thermal properties. They are particularly interested in the variation of the size of the transition zone as model parameters vary. The heat flux on the surface of the blade increases sharply in regions where turbulence begins to develop, but the location and size of the transition zone are very difficult to quantify.

We model the uncertainty in the transition zone by a spatially varying, parameterized model for the heat flux on the surface of the blade geometry. The parameterized model is inspired by Karhunen-Loeve (KL) expansion models for stochastic processes, where the coefficients in a linear combination of spatially varying basis functions are interpreted as random variables with a given distribution; the coefficients in the heat flux model are assumed to vary independently. These independent coefficients comprise the parameters of the heat flux model. The basis functions are approximate eigenfunctions of an exponential correlation kernel scaled by the square root of the corresponding eigenvalues. The correlation length parameters are chosen to admit rapid fluctuations in heat flux along the surface in the streamwise direction and slow variation in the spanwise direction; the rapid fluctuations model the uncertain transition zone.

The quantity of interest is the mean squared value of the computed temperature over the trailing 70% of the blade in the streamwise direction, where turbulence is most likely to occur. The parameters of the heat flux influence the temperature in the blade, which in turn influences the quantity of interest. We thus consider the mean squared temperature over the blade tail as a function of the heat flux parameters to be the high-dimensional function we wish to approximate. We can obtain the derivatives of the quantity of interest with respect to the heat flux parameters by solving the adjoint equation of the forward model as detailed in Section 4.2.

4.1. Forward problem. The heat transfer model is a steady, nonlinear partial differential equation, where nonlinearity arises because thermal conductivity is a function of temperature. Let $\mathcal{S} \subset \mathbb{R}^3$ with elements $\mathbf{s} = (s_1, s_2, s_3)$ represent the spatial geometry shown in Figure 5.5. The spatial domain comes from a NACA0018 vertically symmetric airfoil with three interior holes; the domain is scaled to length 1 in s_1, length 0.2 in s_2, and length 2 in s_3. We seek a temperature distribution $T = T(\mathbf{s})$ that satisfies

$$\begin{aligned} -\nabla_{\mathbf{s}} \cdot (\kappa(T)\, \nabla_{\mathbf{s}} T) &= 0 \quad \text{on } \mathcal{S}, \\ T &= 0 \quad \text{on } \mathcal{B}_1, \\ -\mathbf{n} \cdot (\kappa(T)\, \nabla_{\mathbf{s}} T) &= g \quad \text{on } \mathcal{B}_2. \end{aligned} \tag{4.1}$$

The thermal conductivity takes the form $\kappa = \kappa_0 + \kappa_1 T$, where we choose $\kappa_0 = 0.1$ and $\kappa_1 = 0.01$. The boundary $\mathcal{B}_1$ represents the interior cooling holes, and the homogeneous Dirichlet boundary conditions model a prescribed temperature. The boundary $\mathcal{B}_2$ is the external surface of the blade, and the Neumann boundary condition $g = g(\mathbf{s}, \mathbf{x})$ models the uncertain heat flux, where $\mathbf{x}$ are the model parameters. The function g takes the form

$$g(\mathbf{s}, \mathbf{x}) \;=\; \mu(s_1) \;+\; \gamma(s_1) \sum_{i=1}^{m} x_i\, \phi_i(s_1, s_3). \tag{4.2}$$

The parameters $\mathbf{x} = [x_1, \ldots, x_m]^T$ are independent and uniformly distributed in the hypercube $[-3, 3]^m$. The $\phi(s_1, s_3)$ are products of eigenpairs for the correlation kernel $K : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ given by

$$K\left(\left(s_1^{(1)}, s_3^{(1)}\right), \left(s_1^{(2)}, s_3^{(2)}\right)\right) \;=\; \exp\left(-\theta_1^{-1}\left|s_1^{(1)} - s_1^{(2)}\right| - \theta_3^{-1}\left|s_3^{(1)} - s_3^{(2)}\right|\right). \tag{4.3}$$

Note that the basis functions are only two-dimensional with variation in streamwise and spanwise directions; the correlation length parameters are subscripted to correspond to spatial coordinates. We choose $\theta_1 = 0.01$ and $\theta_3 = 0.1$ to model rapid streamwise and slow spanwise fluctuations. The mean function $\mu = \mu(s_1)$ and scaling function $\gamma = \gamma(s_1)$ are chosen so that $\mu + 5\gamma$ and $\mu - 5\gamma$ are prescribed to cover the heat flux ranges between turbulent and laminar flows; these are shown in Figure 5.4. Figures 5.3a-5.3c show three realizations of heat flux on the turbine blade for three different sets of the parameters.

The temperature $T = T(\mathbf{s}, \mathbf{x})$ depends on the parameters $\mathbf{x}$ of the heat flux. The quantity of interest is expressed as

$$f(\mathbf{x}) \;=\; \int_{\mathcal{S}} T^2\, \mathcal{I}(s_1 > 0.3)\, d\mathbf{s}, \tag{4.4}$$

where $\mathcal{I}(s_1 > 0.3)$ is an indicator function that returns 1 on the trailing region of the blade.

4.2. Adjoint equation and derivatives. To obtain derivatives of f with respect to the heat flux parameters $\mathbf{x}$, we first solve the adjoint equation, which takes the forward solution as inputs. The nonlinearity in the forward model causes the adjoint equation to be an advection-diffusion equation, where the velocity field is the gradient of the computed temperature. Also, the nonlinear quantity of interest causes a forcing term in the adjoint equation that depends on temperature. We do not derive

the adjoint equation, but instead refer interested readers to [5, 19]. We seek a function $Q = Q(\mathbf{s})$ that satisfies

$$\begin{aligned} -\nabla_{\mathbf{s}} \cdot (\kappa(T)\, \nabla_{\mathbf{s}} Q) + \kappa_1 \nabla_{\mathbf{s}} T \cdot \nabla_{\mathbf{s}} Q &= 2\, T\, \mathcal{I}(s_1 > 0.3) \quad \text{on } \mathcal{S} \\ Q &= 0 \quad \text{on } \mathcal{B}_1 \\ -\mathbf{n} \cdot (\kappa(T)\, \nabla_{\mathbf{s}} Q) &= 0 \quad \text{on } \mathcal{B}_2 \end{aligned} \tag{4.5}$$

With the computed solution Q, the partial derivatives of f can be computed as

$$\frac{\partial f}{\partial x_i} \;=\; -\int_{\mathcal{B}_2} Q\, \gamma\, \phi_i \, d\mathbf{s}, \tag{4.6}$$

where γ and ϕ_i are defined in (4.2).

4.3. Solvers and computations. The solutions to the forward and adjoint equations are approximated with a piecewise linear Galerkin finite element method (FEM), where we use streamwise upwind Petrov-Galerkin [14] to stabilize the adjoint equation. We implemented these methods using the Python interface for the FEniCS/DOLFIN suite of tools [26], where the weak forms of the equations are easily expressed in the unified form language (UFL [1]). Through the DOLFIN interface, we employ the PETSc [3] Scalable Nonlinear Equations Solvers with a line search Newton method for the discretized forward problem and the PETSc LU direct solver for the discretized adjoint problem.

The geometry and mesh are constructed with Gmsh [15]. The final mesh contains 32076 nodes. Using the piecewise linear FEM, we thus have 32076 degrees of freedom in both the nonlinear forward problem and the linear adjoint problem. We visualize the solutions using Paraview [18].

The bases for the heat flux in (4.2) are computed using our Random Field Simulation package [9] for Matlab. This package uses the Matlab command `eigs`, which calls ARPACK [23] to compute approximate eigenpairs. These are computed on a two-dimensional grid with 101 nodes in the direction corresponding to streamwise and 31 nodes in the direction corresponding to spanwise. The computed functions are interpolated onto the upper and lower blade surfaces, where the heat flux is defined.

The computations were performed on a Dell Precision workstation with dual quad core processors and 12 GB of RAM. The DOLFIN library is written so that the simulations can run in parallel using all eight cores with a simple `mpirun` call. With all eight cores, each forward problem took approximately 0.4 minutes and each adjoint problem plus derivative computations took approximately 0.8 minutes. However, these codes were not optimized for performance, even within the DOLFIN framework. The solvers with mesh and inputs can be downloaded from `https://github.com/paulcon/fenics-blade`.

4.4. Dimension reduction study. Next we apply the dimension reduction method to the quantity of interest f from (4.4). All computations in this section are done with custom Matlab scripts on the machine described previously; the scripts can be downloaded from `www.stanford.edu/~paulcon`.

The model for the heat flux is characterized by $m = 250$ parameters. Often m is chosen to capture some proportion of the energy represented by the eigenvalues of the correlation kernel. We plot the eigenvalues for our case in Figure 5.6a, where it is apparent that the decay is very slow. To estimate the energy retained with 250 terms, we compute the first 500 out of the 3131 eigenvalues and fit an exponentially

decaying model. From this computation, we have retained approximately 91% of the energy in the field. However, $m = 250$ was chosen more for practical considerations of the computations.

Given m, we first obtain $M = 750$ initial design sites from the parameter space $[-3, 3]^m$. To do this, we sample sets of M m-dimensional points uniformly from the parameter space until we obtain a design whose minimum distance between points is relatively large; this heuristic is inspired by *maximin* designs [22]. For each design site, we compute the heat flux and solve (i) the forward problem to get the quantity of interest and (ii) the adjoint problem to get the gradient.

Next we compute the SVD of the matrix of gradient samples and examine the singular values. The matrix of gradient samples has dimension 250×750, so the SVD is trivially computable on the workstation. The squared singular values are plotted in Figure 5.6a and compared to the eigenvalues of the correlation kernel. This comparison is only meaningful in the sense of dimension reduction. A common strategy for these sorts of problems is to reduce the dimension by truncating the KL expansion according to its eigenvalues. In this case, the slow decay suggests very little reduction is possible. However, the decay of the squared singular values from the gradient samples suggests that substantial reduction is possible for this particular quantity of interest.

Figure 5.6b plots the components of the first two eigenvectors that define the one-dimensional and two-dimensional active subspace. Notice that many components are nearly zero, which suggests that the quantity of interest is relatively insensitive to the corresponding parameters. Figures 5.7a-5.7b show the initial M samples of f projected onto the one-dimensional and two-dimensional active subspaces. Already clear trends appear in the data when their coordinates are projected onto the proper subspaces.

For the one-dimensional projection, we use five nodes on the reduced domain. For the two-dimensional projection, a we place grid of 7×7 points on the bounding box and follow the procedure described in Section 3.2.2 to compute the training data. The pruning procedure left 25 nodes on the two-dimensional subspace. Beyond the initial M samples, we compute f 25 more times for the training data. Figures 5.8a-5.8b and 5.9a-5.9b show one-dimensional and two-dimensional kriging response surfaces, respectively, with training data and prediction variances.

4.5. Comparison with other dimension reduction strategies. We compare the kriging surface on the active subspace to two other approaches that include comparable dimension reduction: (i) a quadratic-mean kriging surface on a one/two-dimensional subspace, where the components are the first/second parameters x_1 and x_2 corresponding to the largest energy modes in the heat flux model, and (ii) a quadratic-mean kriging surface on a one/two-dimensional subspace, where the components are the largest absolute values of the gradient at the nominal parameters $\mathbf{x} = 0$. These are standard dimension reduction strategies. In both cases, maximum likelihood methods were used to tune a correlation length parameter in a Gaussian correlation for the kriging surface.

We use the set of function evaluations f_j generated while computing the gradient samples as a test set of size $M = 750$. With this test set, we compute the error for each response surface method. The statistics of these errors are shown in Table 4.1, and normalized histograms of the errors are shown in Figures 5.10a-5.10b. It is clear that the dimension reduction method based on the active subspace defined by samples of the gradient is superior for this problem.

$n = 1$	ASM	KL	SENS
max. err.	7.5639e-03	3.4637e-01	3.2485e-01
mean err.	1.9846e-03	7.6402e-02	7.0928e-02
$n = 2$	ASM	KL	SENS
max. err.	6.7920e-03	3.3185e-01	2.4453e-01
mean err.	1.8801e-03	6.7964e-02	6.0523e-02

Table 4.1: The error of the reduced dimension response surfaces over the testing set. Methods labeled KL use the first parameters from the Karhunen-Loeve-type model for the heat flux boundary (4.2). Methods labeled SENS use the parameters with the largest absolute value of the gradient $\nabla_{\mathbf{x}} f$ at the origin. Methods labeled ASM use the active subspace defined by the first two eigenvectors of the matrix $\tilde{\mathbf{C}}$ in (3.2). Each method is tested in one and two dimensions.

5. Summary & conclusions. Active subspace methods seek to approximate a multivariate function on a low-dimensional subspace of its domain that contains the majority of the function's variability. We have analyzed a theoretical best approximation on the active subspace and derived error bounds. We have used these analyses to motivate a computational procedure for detecting the directions defining the subspace and constructing a kriging response surface on the subspace to approximate the function over its domain. We have applied this procedure to nonlinear heat transfer model with a 250-parameter model of uncertain heat flux on the boundary. We compared the active subspace method to two other approaches for dimension reduction with striking success.

Loosely speaking, active subspace methods are appropriate for certain classes of functions that vary primarily in low-dimensional subspaces of the input. If there is no decay in the eigenvalues of $\mathbf{C}$, then the methods will perform poorly; constructing such functions is not difficult. However, we have found many high-dimensional applications in practice where the eigenvalues do decay quickly, and the functions respond well to active subspace methods [7, 12, 10, 11]. Most of those applications look similar to the one presented in Section 4, where uncertainty in some spatially varying physical input can be represented by a series expansion, and the coefficients of the expansion are treated as random variables; such models arise frequently in UQ.

The computational method we have proposed is ripe for improvements and extensions. We have mentioned many such possibilities in Section 3.3, and we are particularly interested in methods for using fewer evaluations of the gradient to compute the directions defining the active subspace. We will also pursue strategies that make better use of the function evaluations acquired during the gradient sampling.

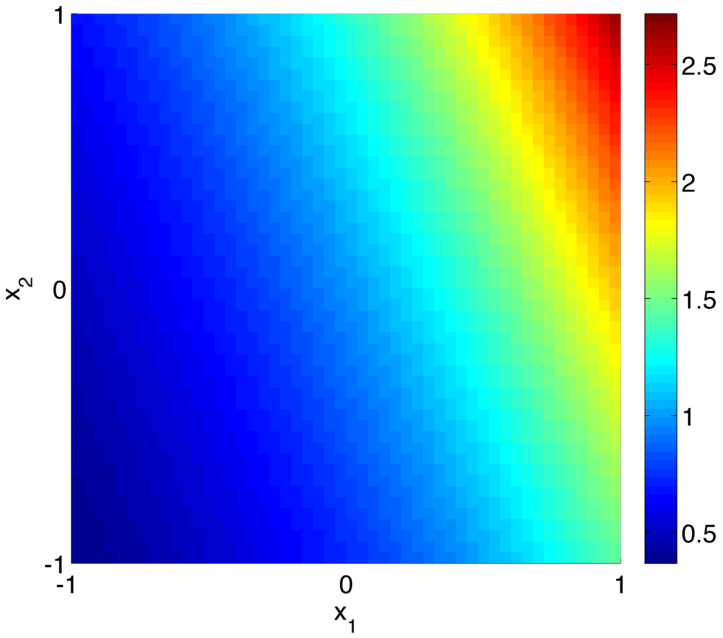
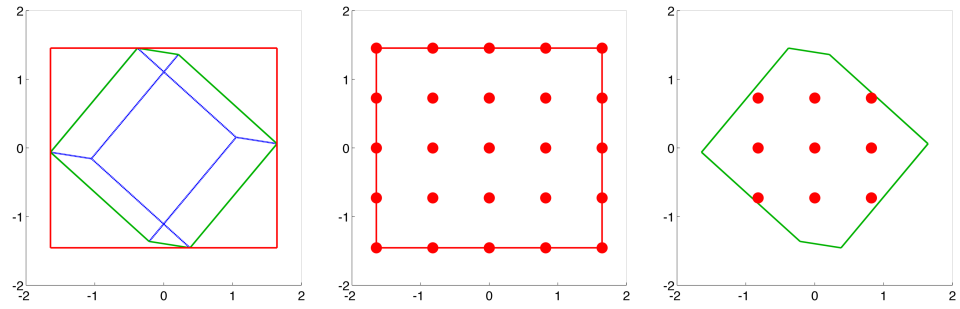
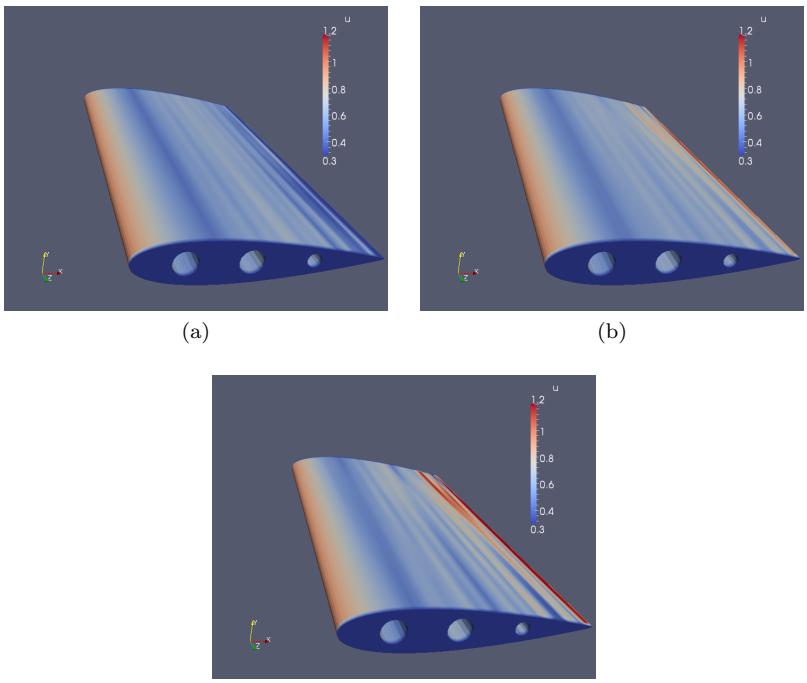


Fig. 5.1: The function $f(x_1, x_2) = \exp(0.7x_1 + 0.3x_2)$ varies along the direction $[0.7, 0.3]^T$, and it is flat in the direction $[0.3, -0.7]^T$. (Colors are visible in the electronic version.)



(a) (b) (c)

Fig. 5.2: (5.2a) A three dimensional cube (blue) projected onto a two dimensional plane. The green lines represent the convex hull of the projected corners of the cube. The red lines show the bounding box of the convex hull in two dimensions. (5.2b) A design on the active subspace. Note that not every point in this design corresponds to a point in the original space. (5.2c) The final design for the active subspace is the set of points in the grid for which there exists a point in the full space. Note that we do not compute the convex hull in practice. (Colors are visible in the electronic version.)



(a) (b)

(c)

Fig. 5.3: Three realizations of the heat flux (4.2) on the surface of the turbine blade. (Colors are visible in the electronic version.)

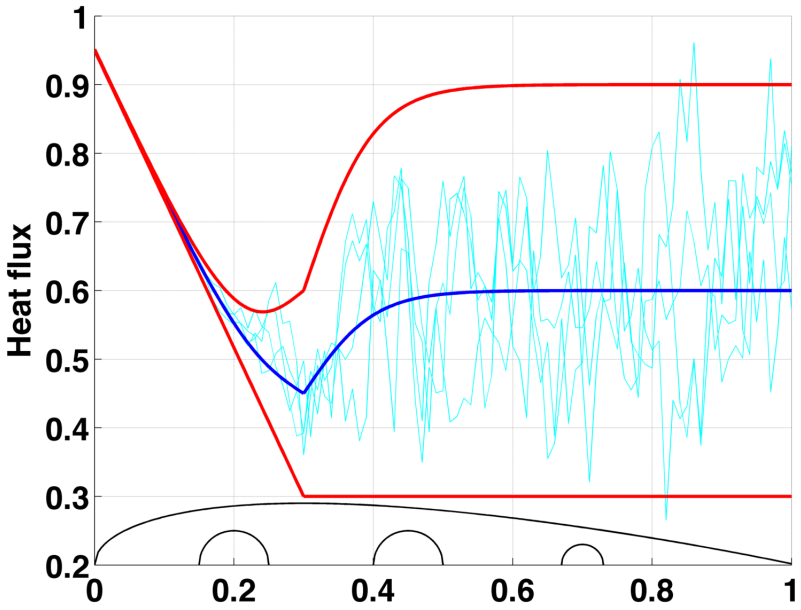


Fig. 5.4: The mean μ and scaling γ functions of the Karhunen-Loeve-type model for the heat flux (4.2). These functions are crafted to admit heat flux realizations that represent a transition to turbulence of the flow over the blade. The mean μ is shown in blue. The fluctuation $\mu \pm 5\gamma$ is shown in red. Five realizations of the heat flux down the spanwise centerline of the blade are shown in cyan. The upper surface of the blade is shown in black for reference. (Colors are visible in the electronic version.)

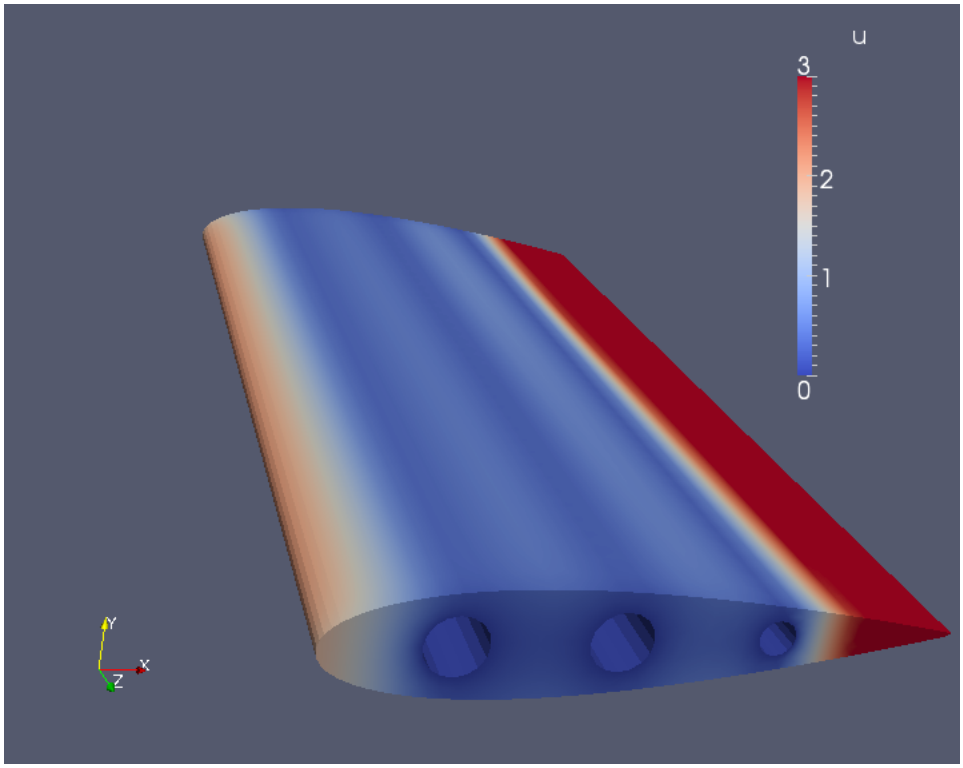
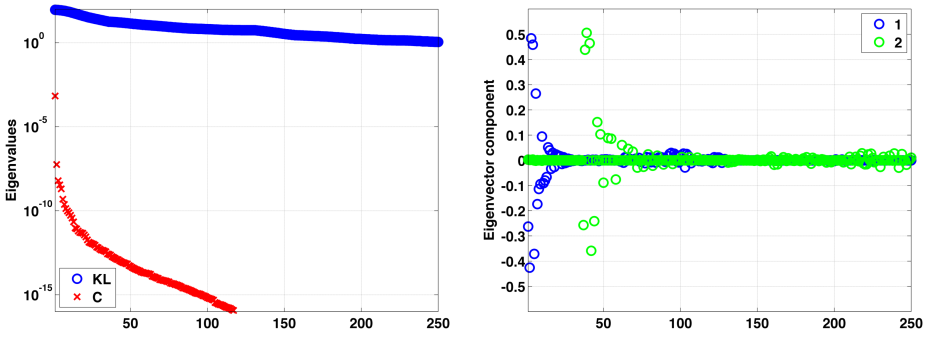
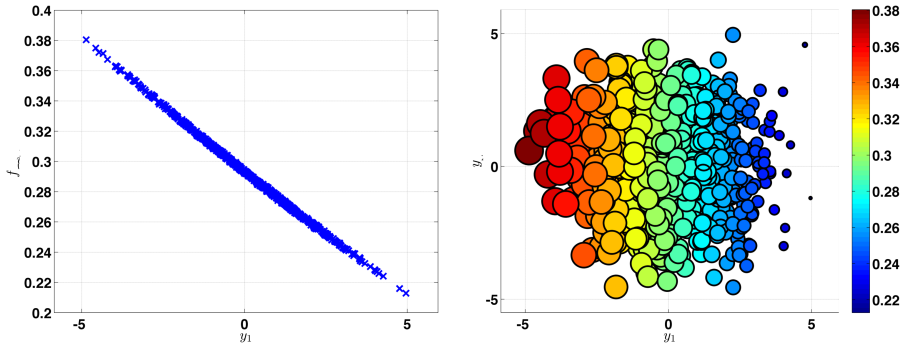


Fig. 5.5: The temperature distribution for one set of heat flux parameters. (Colors are visible in the electronic version.)



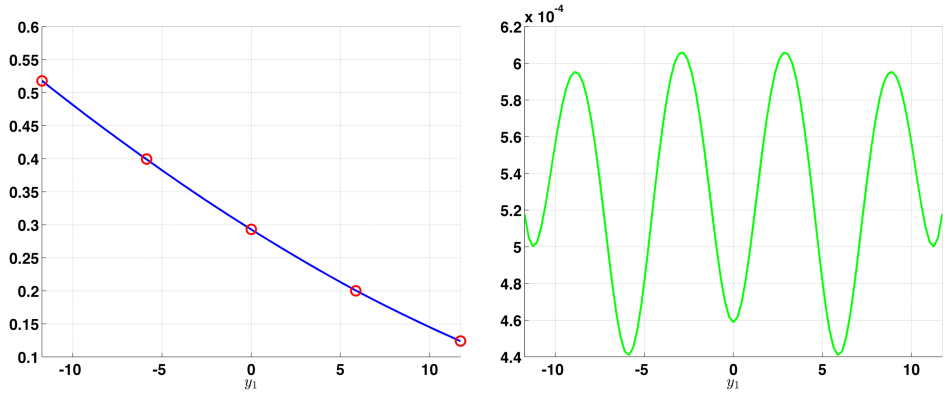
(a) Eigenvalues (b) Eigenvectors

Fig. 5.6: (5.6a) Eigenvalues of the Karhunen-Loeve like expansion defining the heat flux (see (4.2)) compared to the eigenvalues of the matrix $\tilde{\mathbf{C}}$ from (3.2). (5.6b) Components of the first two eigenvectors of $\tilde{\mathbf{C}}$ from (3.2) that define the active subspace. (Colors are visible in the electronic version.)



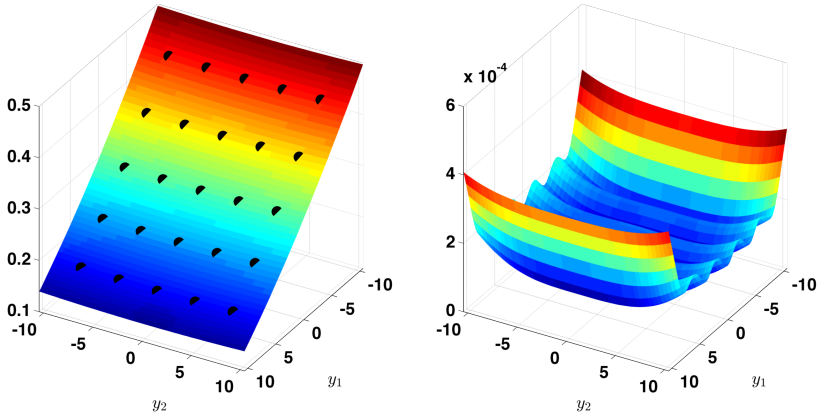
(a) 1d projection (b) 2d projection

Fig. 5.7: Function values $f_j = f(\mathbf{x}_j)$ for $\mathbf{x}_j \in [-3, 3]^m$ projected onto the one-dimensional (5.7a) and two-dimensional (5.7b) active subspace. (Colors are visible in the electronic version.)



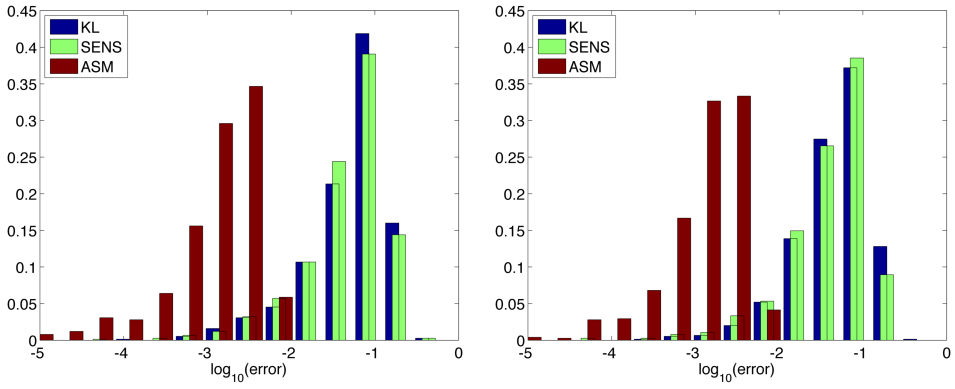
(a) Kriging prediction (b) Kriging std.

Fig. 5.8: Kriging approximation $\tilde{f}_n(\mathbf{y})$ and its standard deviation on the one-dimensional active subspace. (Colors are visible in the electronic version.)



(a) Kriging prediction (b) Kriging std.

Fig. 5.9: Kriging approximation $\tilde{f}_n(\mathbf{y})$ and its standard deviation on the two-dimensional active subspace. (Colors are visible in the electronic version.)



(a) One-dimensional (b) Two-dimensional

Fig. 5.10: Comparison of errors for subspace projections. One-dimensional projections are in Figure 5.10a, and two-dimensional projections are in Figure 5.10b. "KL" refers to projecting onto the first two parameters of the heat flux model. "SENS" refers to projecting onto the parameters with the two largest components of the gradient. "ASM" is the projection onto the two-dimensional active subspace. (Colors are visible in the electronic version.)

REFERENCES

[1] M. S. Alnaes, A. Logg, K. B. Oelgaard, M. E. Rognes, and G. N. Wells, *Unified form language: A domain-specific language for weak formulations of partial differential equations*, arXiv preprint arXiv:1211.4047, (2012).

[2] A. C. Antoulas, *Approximation of large-scale dynamical systems*, vol. 6, Society for Industrial and Applied Mathematics, 2005.

[3] S. Balay, J. Brown, , K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, *PETSc users manual*, Tech. Rep. ANL-95/11 - Revision 3.3, Argonne National Laboratory, 2012.

[4] M. Bebendorf, *A note on the poincaré inequality for convex domains*, ZEITSCHRIFT FUR ANALYSIS UND IHRE ANWENDUNGEN, 22 (2003), pp. 751–756.

[5] A. E. Bryson and Y.-C. Ho, *Applied Optimal Control: Optimization, Estimation, and Control*, Hemisphere Publishing Corportation, 1975.

[6] J.-F. Cai, E. J. Candes, and Z. Shen, *A singular value thresholding algorithm for matrix completion*, SIAM Journal on Optimization, 20 (2010), pp. 1956–1982.

[7] H. Chen, Q. Wang, R. Hu, and P. Constantine, *Conditional sampling and experiment design for quantifying manufacturing error of a transonic airfoil*, AIAA-2011-658, (2011).

[8] A. Cohen, I. Daubechies, R. DeVore, G. Kerkyacharian, and D. Picard, *Capturing ridge functions in high dimensions from point queries*, Constructive Approximation, pp. 1–19. 10.1007/s00365-011-9147-6.

[9] P. G. Constantine and Q. Wang, *Random field simulation*, 2010.

[10] P. G. Constantine, Q. Wang, A. Doostan, and G. Iaccarino, *A surrogate-accelerated Bayesian inverse analysis of the HyShot II flight data*, AIAA-2011-2037, (2011).

[11] P. G. Constantine, Q. Wang, and G. Iaccarino, *A method for spatial sensitivity analysis*. Center for Turbulence Research, Annual Brief, 2012.

[12] E. Dow and Q. Wang, *Output based dimensionality reduction of geometric variability in compressor blades*, AIAA-2013-0420, (2013).

[13] M. Fornasier, K. Schnass, and J. Vybiral, *Learning functions of few arbitrary linear parameters in high dimensions*, Foundations of Computational Mathematics, 12 (2012), pp. 229–262.

[14] L. P. Franca, S. L. Frey, and T. J. Hughes, *Stabilized finite element methods: I. application to the advective-diffusive model*, Computer Methods in Applied Mechanics and Engineering, 95 (1992), pp. 253–276.

[15] C. Geuzaine and J.-F. Remacle, *Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities*, International Journal for Numerical Methods in Engineering, 79 (2009), pp. 1309–1331.

[16] A. Griewank, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, SIAM, 2000.

[17] N. Halko, P.-G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM review, 53 (2011), pp. 217–288.

[18] A. Henderson, J. Ahrens, and C. Law, *The ParaView Guide*, Kitware Clifton Park, NY, 2004.

[19] A. Jameson, *Aerodynamic design via control theory*, Journal of scientific computing, 3 (1988), pp. 233–260.

[20] I. Jolliffe, *Principal Component Analysis*, Springer Verlag, 2nd ed., 2002.

[21] D. R. Jones, *A taxonomy of global optimization methods based on response surfaces*, Journal of global optimization, 21 (2001), pp. 345–383.

[22] J. Koehler and A. Owen, *Computer experiments*, Handbook of statistics, 13 (1996), pp. 261–308.

[23] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*, vol. 6, Siam, 1998.

[24] J. Li and D. Xiu, *Evaluation of failure probability via surrogate models*, Journal of Computational Physics, 229 (2010), pp. 8966–8980.

[25] C. Lieberman, K. Willcox, and O. Ghattas, *Parameter and state model reduction for large-scale statistical inverse problems*, SIAM Journal on Scientific Computing, 32 (2010), pp. 2523–2542.

[26] A. Logg, K.-A. Mardal, and G. Wells, *Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book*, vol. 84, Springer, 2012.

[27] D. G. Luenberger and Y. Ye, *LInear and Nonlinear Programming*, Springer, 2008.

[28] Y. M. Marzouk, H. N. Najm, and L. A. Rahn, *Stochastic spectral methods for efficient*

Bayesian solution of inverse problems, Journal of Computational Physics, 224 (2007), pp. 560–586.

[29] B. Nadler, *Finite sample approximation results for principal component analysis: A matrix perturbation approach*, The Annals of Statistics, 36 (2008), pp. pp. 2791–2817.

[30] A. Owen, *Variance components and generalized sobol' indices*, SIAM/ASA Journal on Uncertainty Quantification, 1 (2013), pp. 19–41.

[31] R. Pecnik, J. A. Witteveen, and G. Iaccarino, *Assessment of uncertainties in modeling of laminar to turbulent transition for transonic flows*, Flow, Turbulence and Combustion, (2013), pp. 1–21.

[32] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*, vol. 1, MIT press Cambridge, MA, 2006.

[33] T. M. Russi, *Uncertainty Quantification with Experimental Data and Complex System Models*, PhD thesis, UC Berkeley, 2010.

[34] Y. Saad, *Numerical methods for large eigenvalue problems*, vol. 158, SIAM, 1992.

[35] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola, *Global sensitivity analysis: the primer*, Wiley-Interscience, 2008.

[36] H. Wendland, *Scattered data approximation*, vol. 2, Cambridge University Press Cambridge, 2005.

[37] D. Williams, *Probability with martingales*, Cambridge university press, 1991.

[38] K. Zhou, J. C. Doyle, K. Glover, et al., *Robust and optimal control*, vol. 40, Prentice Hall, 1996.